Tracking ID: 580
A Simple Protocol for Coordinating Planning Agents

Bradley J. Clement and Anthony C. Barrett
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109-8099
{bclement, barrett}@aig.jpl.nasa.gov

ABSTRACT
Planning and scheduling systems offer efficient ways of determining a course of action for an agent to meet its goals in the presence of complex resource constraints. Decentralized coordination technology has formulated these problems in terms of constraint satisfaction, but protocols for planning agents have not been significantly developed. Interacting space missions require decentralized coordination of mission planning systems. This paper introduces a simple decentralized protocol that enables separate planners to coordinate shared tasks and resolve conflicts over shared and private resources. This protocol is independent of the nature of the planners it coordinates. The protocol is applied to a simulated Mars 2003 scenario involving two Mars Exploration Rovers and the Odyssey orbiter. This application reveals many research challenges for coordinating planning agents.

Keywords
coordinating multiple agents & activities, conflict resolution and negotiation, action selection and planning, coordination protocols, coordination applications

1. INTRODUCTION
Planning and scheduling systems offer efficient ways of determining courses of actions for agents to meet their goals in the presence of complex resource constraints. Decentralized coordination technology has formulated these problems in terms of constraint satisfaction problems [16, 10], but protocols for planning agents have not been significantly developed.[1] Decentralized coordination is necessary when agents are reluctant to disclose private information, must negotiate over resources in contention in competitive environments, or receive local information that cannot be efficiently shared with others.

[1]Related coordination and planning work [6, 5] is discussed in Section 5.

AAMAS 2001, Bologna Italy

For example, there is currently a trend toward multi-spacecraft missions and greater interaction of separate missions. Interacting space missions require coordination over shared tasks and resources. Each mission has its own goals to collect scientific data from its spacecraft (probe, orbiter, or surface explorer). The scientists and operation staff of one mission must coordinate with those of other missions to resolve any conflicts by potentially making concessions in the efficiency or goals of their plan. Costs for ground operations must be minimized, and automated planning and scheduling systems can reduce this cost as has been demonstrated for the Modified Antarctic Mapping Mission. However this is not enough for interacting missions that require a lot of coordination among mission teams.

NASA as well as other space programs in Europe plan to send several orbiters and surface explorers to Mars in the next decade. Mars Global Surveyor and Odyssey are already orbiting Mars; two Mars Exploration Rovers, the Mars Express orbiter, and the Beagle 2 lander are scheduled to launch in 2003; Mars Reconnaissance Orbiter is expected to launch in 2005; and three or more missions, some involving teams of spacecraft are planned for 2007 with others planned in subsequent years. Within this context, surface explorers will depend on orbiters for relaying data, and coordinated measurements are expected among surface exploring teams and potentially with orbiters to maximize science return. Each of these missions will be burdened with their own operations planning and cannot afford the extra overhead of manually coordinating their plans with others. The development of mission plans is naturally decentralized, and their coordination must be automated in order to scale up to the expected explosion of interactions. The coordination cannot be accomplished centrally because each mission needs control over its own planning process.

This paper introduces a simple decentralized protocol that enables separate planners to coordinate shared tasks and resolve conflicts over shared and private resources. By restricting the set of tasks that are shared between agents, the constraints associated with them, and the ability to modify certain kinds of shared tasks, a domain modeler can construct a tailored coordination strategy on top of our simple protocol. This protocol is independent of the nature of the planners it coordinates. We apply this concept in a simulated Mars 2003 scenario to coordinate two Mars Exploration Rovers (MERs) and the Odyssey orbiter whose operations staff must coordinate over the transmission of scientific data and the reception of uplinks from Earth (Figure 2). The mission plans for each spacecraft involve over a thousand activities and tens of resource and state variables. Each planner is an ASPEN process. ASPEN is an instance of a heuristic iterative repair planner/scheduler that can

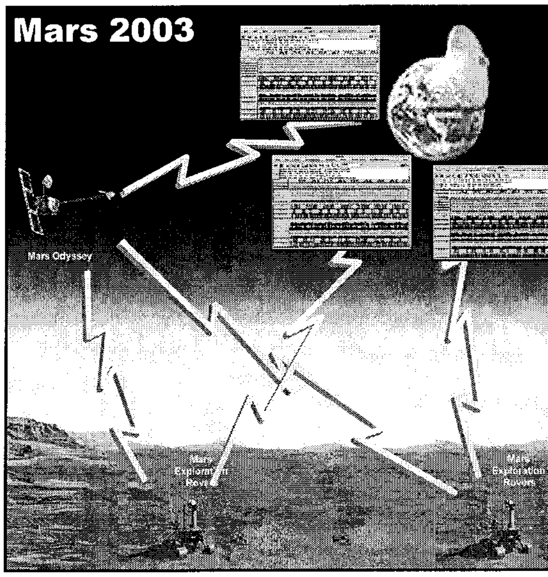

Figure 1: Ground planners coordinate Mars Exploration Rovers and Odyssey

generate consistent plans for agents involving large numbers of activities with constraints on complex resources [1].

We describe the Mars scenario in the following section, and describe the coordination protocol in Section 3. We then describe in Section 4 how ASPEN models and generates plan for the spacecraft, how the ASPEN processes use the protocol to coordinate over data transmission, and how the protocol enables the mission planners to quickly recover from an unexpected failure. Next we discuss how this approach differs from other work in plan coordination in Section 5. Finally, we summarize our contributions and describe research questions raised by this work.

2. MARS 2003 SCENARIO

Currently, Odyssey is orbiting Mars and will begin its more than two year mission of mapping the surface of Mars by analyzing the mineral content of the surface while searching for water and by measuring radiation to help us understand the implications of possible manned missions. In 2003, two Mars Exploration Rovers (MERs) are scheduled to launch and land on different locations on the Mars surface. These rovers will be equipped with several instruments for imaging their surrounding area, detecting water, and analyzing the composition and textures of rocks on a microscopic scale. They can travel as far as 100 meters in one Martian day.

Odyssey's core mission will end around the time that the MERs arrive, and it will assist the rovers in relaying data to and from Earth. However, Odyssey will have completely mapped only a fraction of the planet's surface, and scientists will undoubtedly wish to continue gathering data. The MERs (MER A and MER B) can communicate directly to Earth at low bandwidth when it is in view from the Martian surface once per day for several hours, but communication requires a large amount of power, the rovers' most constrained resource. Each MER has the opportunity to communicate with Odyssey for five to ten minutes on each of two passes per day (because of its polar orbit) at much higher bandwidth and with less power requirements. Odyssey can communicate with Earth at low bandwidth (but roughly three times greater than the rovers with Earth) every other hour for an hour when Earth is in view. The bandwidth to Earth is restricted such that only a small fraction of scientific data can be returned to Earth. Thus, optimizing the routing of data can significantly improve science return. In addition, it can mean the difference in whether a rover receives new commands from Earth in time to execute them.

Each MER's mission is planned on a daily cycle. Using data describing the state of the MER, the scientists and the operations team generate a schedule for the next day and uplink the commands either directly to the MER or through Odyssey. The MER performs minimal activities at night to preserve its battery energy when solar energy is not available and then executes the uplinked commands during daylight. The rover sends data back to Earth either directly or through Odyssey with information to help the mission team generate the next day's plan. These plans are generated on a daily basis because the MER's environment and actions are uncertain, and anomalies are expected that can cause the rover to fail to accomplish its goals. In contrast, Odyssey's plans are generated on a weekly cycle since the environment is more controlled.

Suppose that the mission planners continually update three-day plans for each rover even though only a day's commands will be uplinked. MER A performs science experiments at its current location on day 1, performs more on day 2, but also travels to a new location at the end of the day to prepare for experiments on day 3. MER B travels at the beginning of day 1, performs experiments the same day, performs more on day 1, and travels at the end of day 3 after further experiments. Critical image data is taken by these rovers each day before traveling. The rovers need to downlink the data to Earth and receive an uplinked response from Earth in time to direct the traverse to the chosen location.

The coordination problem is that the mission planners need to agree on how the data is to be downlinked and uplinked. If data is routed through Odyssey, Odyssey must not only schedule the communication activities but also confirm that it has the memory storage, power, battery energy, and time needed to perform the communication. Data of different criticality is often relayed from the MERs through Odyssey, so Odyssey must decide in what order it will send the data. To accommodate the rovers, Odyssey may need to delete data, reschedule its own measurements, or even throw them out. Of course, this could be a problem for the Odyssey team if valuable, time-dependent measurements are missed. At the same time, if a rovers critical data is delayed, ground operations may not have time to generate the next day's plan, and valuable time will be lost while the rover remains idle. Thus, the mission teams need to negotiate over how to resolve their conflicts.

By automating this coordination, human operations cost is reduced, responses to anomalies and opportunities are sooner uplinked, and human errors can be avoided. In Section 4, we describe how mission planning systems for ground operations coordinate using the protocol described in the next section.

3. COORDINATION PROTOCOL

As argued by Decker in [5], no general coordination strategy is best for a particular domain. The protocol we describe here, however, is general enough to capture a large space of of strategies. The content of the messages passed between agents is simply *shared tasks* with constraints on timing and state/resource variables. Our protocol is ignorant of the representation and semantics of the task and the planners that use them, but tasks are commonly sets of constraints (conditions and effects) on states and resources and potentially have timing constraints such as duration, start times, and/or ordering relations with other tasks. A task is *shared* when it has constraints represented in more than one agent's plan. For example, MER B may have a shared communication task with Odyssey to transmit some amount of data of a particular criticality at noon on day 2 for six minutes. Shared tasks may only be partially specified. MER B does need to send the constraints on local resources, such as MER B's power, and MER B does not have to tell Odyssey its constraints on its local power resource–Odyssey interprets the shared task to have different constraints on power, energy, and memory storage based on the timing, duration, and amount of data sent with the task.

The protocol also allows hierarchical tasks with decompositions to be passed at multiple levels of abstraction. A decomposition of a task in one agent's plan may be updated in another's plan by passing the subtasks, or the receiving agent can have a different decomposition of the shared task.

So shared tasks can create conflicts in the plan of one agent or in all the agents' plans. The protocol allows an agent to only send tasks from a *shared set* as determined by the domain modeler.[2] Each shared task in the set is a schema or type of task as opposed to particular instances of the task. We limit agents to only be able to modify a task of its own and not any that are passed by another agent unless it appears in both agents' shared sets. Modifications could include changing constraints, rescheduling the task, or removing the task. For example, if both MER B and Odyssey have a shared communication task in their sets, then either may propose or delete those task instances. However, if the communication task is only in MER B's shared set, then the Odyssey planner is forced to adapt its plan to accommodate the communication task.

So, if a task schema is in one agent's shared set and not in another's, then an instance is a *command task* because it is the recipient's responsibility to resolve conflicts in its own schedule. A instance of a task schema in the shared sets of a group of agents is a *coordination task* because the group must coordinate over the scheduling of the task. When an agent creates an instance of a coordination task, it is a *request* for the other agents in the group to agree upon. Shared tasks communicate constraints but can be used to only propagate local state information to other interested parties. In this sense, these tasks can be used to communicate and coordinate beliefs, intentions, and goals. This characterization ties in work focused on BDI (belief-desire-intention) models of agents [14] and shared mental states (shared plans) of collaborative agents [9].

If a state or resource is only affected by constraints of shared tasks common to a group of agents, then it is a *shared state/resource* of the group, and the group must come to consensus on the constraints since all are shared. Otherwise, the state/resource is *local* to the agent because the agent has its own view of the variable's constraints and conflicts.

Given only this specification of shared tasks, the domain modeler has the responsibility of designing the coordination strategy by deciding which tasks schemas belong in which agents' shared sets, what partial information is passed for each task schema for each agent, and how the local constraints of the tasks are specified for each agent. While this puts a great burden on the domain modeler, it enables our simple coordination protocol that can adopt a variety of strategies that adopt simple and complex models of agent mental state.

The protocol of an agent asynchronously alternates between replanning and sending shared task updates while incorporating shared task updates from others. This is depicted in Figure 2. Before coordination, the agents can develop plans independently optimizing their own utility and assuming no negative interactions with others.

[2]A domain modeler is an expert that designs the agent's actions and constraints and specifies which actions will be shared and what partial information is shared.

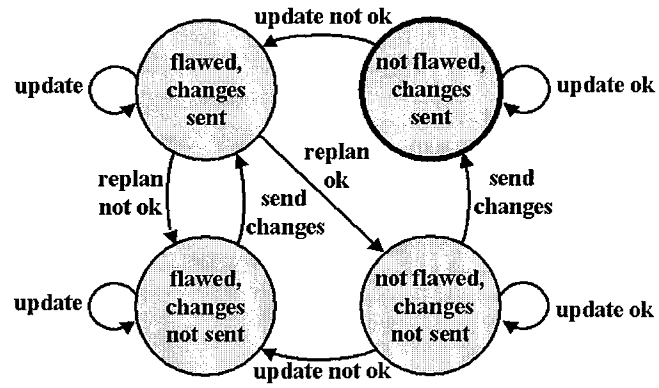

Figure 2: Coordination protocol

The protocol begins with every agent in one of the bottom states in the figure. At the start, shared tasks are sent as updates to other agents' plans as commands or requests unless an update is first received. Any shared task updates from other agents are incorporated into the plan. Then, the agents *replan* for a specified amount of time to resolve any conflicts and optimize with respect to others actions. This replanning can involve making modifications to resolve conflicts over shared and local states/resources in a local search planner. A refinement planner can continue its search with the added plan information by further elaborating actions in reaction to any added tasks from other agents or backtracking to a previous search state where the plan can be rebuilt around the constraints placed by the shared task updates. Completeness of the refinement planner can still be guaranteed if the search space is never pruned. The key is that the planner cannot modify a command task of another agent. However, a coordinated task in its own shared set can be modified. This is straightforward for plan-space planners that always have a partially or fully elaborated plan. For state-space planners, that do a forward or backward expansion and search through action state sequences, expansion must be constrained to include the command tasks and allow the coordinated tasks.

An agent is in a flawed state if there are state/resource conflicts or if the agent is dissatisfied with some aspect (such as utility) of the plan, and the agent is in one of the states on the left in Figure 2. Coordination is complete when all agents are in the top, right state in and no messages are pending. At this point, all shared tasks are updated in all other agents' plans; all shared states/resources have the same constraints; all local and shared states/resources are conflict-free; and all agents are satisfied with their plans. However, the agents must establish consensus that they are all in the top, right state in order to recognize that coordination is complete. Algorithms for establishing consensus depend on the communication characteristics of the network of planners, and is a subfield of distributed systems. We do not discuss these algorithms. At the same time, establishing consensus may not be important in continual coordination settings where agents alter their plans based on the adoption of new goals or on local state updates. In this case, coordination continues because agents move into one of the bottom states in the figure and send shared task updates.

4. APPLICATION TO MARS SCENARIO

Now we describe how this protocol is applied to the Mars 2003 scenario described in Section 2. The mission planners for each of the MERs and Odyssey first build their three-day plans separately to maximize science return without violating any local constraints

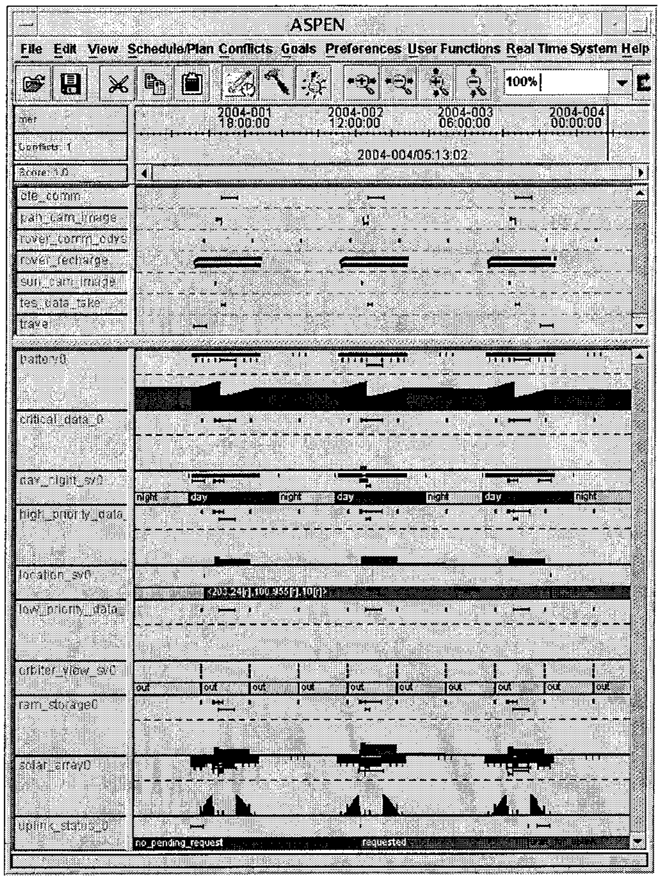

Figure 3: Initial plan for MER B

on memory, power, battery energy, *etc.* Without having to worry about the constraints of the other missions, each mission can more rapidly generate their plans and have their ideal plan as a starting point before making concessions during the coordination process.

In simulating this scenario, we modeled each spacecraft and its plans using ASPEN (Automated Scheduling and Planning ENvironment) [1]. Specific instruments and scientific measurements on the spacecraft were modeled with constraints on solar power, battery energy, memory storage and different priority classes of data, position, antenna usage, *etc.* ASPEN provides a rich language and programming interface for specifying user-defined resources [11] and relationships among resources, constraints, and tasks. The three-day schedules constitute over 600 tasks for each MER and over 1400 for Odyssey with 30 state/resource variables for each MER and 22 for Odyssey. Figure 3 shows a subset of tasks and state/resource timelines in MER B's plan in the ASPEN planner before coordination. Below the menu, toolbar, and time ruler are intervals of scheduled task instances categorized by type (schema). Below the divider are resource and state profiles computed for the schedule. Resources are charted according to their availability/depleted usage. The timeline at the bottom shows a state timeline with state transitions labeled.

The MERs each have their own communication request tasks with Odyssey that conditionally decompose into state changing tasks depending on a shared state for tracking critical data downlink/uplinks. This models the downlink of critical data requiring a subsequent uplink from ground control with new command sequences.[3] These

[3]The actual uplinked commands are not translated into changes in the schedule since the planners are on the ground and the mission team can modify the plans as it wishes, and the commands can simply be parameters to activities such as the new position to which a MER traverses.

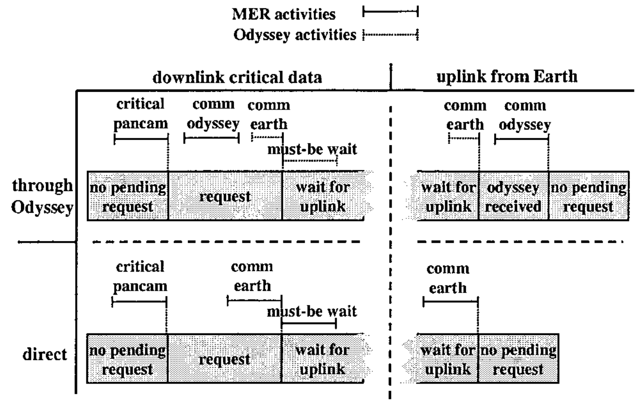

Figure 4: Downlink/uplink states for a rover

communication requests can be with Odyssey or can be command tasks for communicating directly with Earth. The shared state changes are shown in Figure 4. Line segments are the scheduled shared tasks, and blocks are the shared state values over time intervals. Note that Odyssey also incorporates its own state changing request tasks for routing data between a MER and Earth. When the rover collects critical data, it changes the state to `request` to communicate its need to downlink and receive an uplink. Once the data is sent to Earth (either directly or through Odyssey), the state changes to a `wait for uplink` state to indicate it is now just waiting for the uplink. Earth needs a period of time to generate new commands for the uplink, so If the uplink is received by Odyssey, the state changes to `received` to indicate that now the rover can get the uplink from Odyssey. Once the rover receives the uplink, the state changes back to the normal `no pending request` state. Rover tasks (such as a traverse) need the uplinked data before executing, so it places a local constraint that shared state be `no pending request` during its scheduled interval. Another shared state is whether the orbiter is in view of the rover. There are no shared resources although communication requests from a MER have effects on many local resources of both the MER and Odyssey.

When coordination begins, the planners send their communication requests to the other planners. Before these updates are received, the initial views of the shared uplink status are shown in Figure 5. They are in disagreement because the MERs do not have Odyssey's help in relaying critical data, and Odyssey knows of no communication requests from the rovers. The MERs begin with conflicts with their traverse tasks because the uplink has not yet been received from Earth. The coordination protocol command the planners to repetitively process shared task updates, replan to resolve conflicts by recomputing the shared state and modifying scientific measurement operations to adjust for the increased power and memory needs, and send task updates. After a minute and a half, MER A, B, and Odyssey agree on routing the downlink and uplink through Odyssey to get the uplinked commands in time for the traversal on different days.[4] The resultant shared state is shown at the bottom of Figure 5. The ASPENs reach consensus that coordination is complete and sleep while waiting for task updates.

[4]Odyssey's ASPEN process was run on a Sparc Blade, and the MERs ran on a Sparc Ultra 60 and 80.

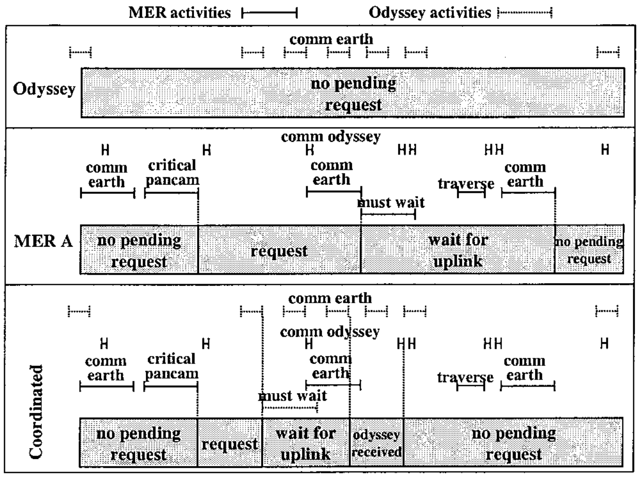

Figure 5: Downlink/uplink shared state for MER A. From top to bottom, Odyssey's initial view, MER A's initial view, and the common view after coordination.

Then we triggered an anomaly in MER A's plan causing it to cancel its first day's tasks and shift the entire schedule forward a day. Before sending the updated shared tasks, replanning was issued to resolve local constraints to avoid propagating inconsistent state information to Odyssey. All conflicts were resolved in a few seconds except the traverse conflicts with a `wait` state. Then MER A sends a task update to restart coordination. Coordination completes in less than a minute with data again being routed through Odyssey.

This application of the coordination protocol to the Mars scenario shows how mission planners can minimize their human coordination efforts during mission execution while making minimal concessions to their ideal plans. Current methods for coordinating spacecraft involves restricting resource availability up front for the spacecraft over the course of their missions so that conflicts do not arise. If the mission planners wish to modify their plans, the mission teams must negotiate and verify conflict resolutions without tools for automating plan adaptations. This coordination tool in conjunction with automated planners enables mission planners to investigate a variety of optional coordination solutions while minimizing human communication and verification effort. In the next section, we discuss how related work does not address decentralized planning coordination problems but offers improvements and strategies that this protocol can exploit. We also discuss new research challenges for decentralized coordination.

5. DISCUSSION AND RELATED WORK

Conflicts among a group of agents can be avoided by reducing or eliminating interactions by localizing plan effects to particular agents [12], and by merging the individual plans of agents by introducing synchronization actions [8]. In fact, planning and merging can be interleaved [7]. Earlier work studied interleaved planning and merging and decomposition in a distributed version of the NOAH planner [3] that focused on distributed problem solving. More recent research builds on all of these techniques by formalizing and reasoning about the plans of multiple agents at multiple levels of abstraction to localize interactions and prune unfruitful spaces during the search for coordinated global plans [2]. While this is a centralized approach, work is needed to apply these techniques that leverage abstraction in a decentralized framework to reduce communication and computation during coordination. Abstract plan information can even automate the discovery of agent relationships that our approach pushes off on the domain modeler.

DSIPE [6] employs a centralized plan merging strategy for distributed planners for collaborative problem solving using human decision support. Like our approach, local and global views of planning problem help the planners coordinate the elaboration and repair of their plans. DSIPE provides insight into human involvement in the planning process as well as automatic information filtering for isolating necessary information to share. While our approach relies on the domain modeler to specify up front what information will be shared, these techniques can be applied in a non-collaborative, fully decentralized framework such as ours.

As mentioned in Section 3, Grosz's shared plans model of collaboration presents a theory for modeling multiagent belief and intention [9]. The differentiations of shared tasks as command, request, and coordination tasks provides basic mechanisms for agents to communicate and establish beliefs, intentions, and goals for itself or a group. Using these constructs to reason about the mental states of agents provides groundwork for the shared plans model and work based on BDI (belief-intention-desire) models of agents [14] to be exploited in competitive domains.

As suggested by the study of Generalized Partial Global Planning [4], a general coordination strategy cannot be best for a variety of problems. This work investigates how to combine different coordination strategies to approach specific domains. Our general protocol offers an alternative framework for separating implementation of these mechanisms from the planning algorithms employed by specific agents. Studying how these and other mechanisms for negotiating over shared resource conflicts can apply in our framework can provide domain modelers with efficient tools to apply to a large variety of coordination problems.

Another issue not addressed by this paper is the potential for agents to repetitively undo each others progress on resolving flaws with shared coordination tasks. Since the agents have the ability in this case to modify the same set of tasks, the modifications they make can clash such that consensus is never established on resolving a common conflict. Additional investigations into more efficient handling of shared conflicts is needed. Distributed constraint satisfaction techniques describe how this can be done for distributed problems formulated as constraint graphs [16, 10]. Adapting these techniques for competitive planning agents may alleviate this problem.

Finally, TEAMCORE provides a robust framework for developing and executing team plans [15, 13]. This work also offers a decision-theoretic approach to reducing communication within a collaborative framework. We expect this work to offer insights into efficient communication and the integration of deliberative planning with robust execution in competitive coordination domains. A related open problem is the continual interleaving of coordination and execution.

6. CONCLUSION

This paper addresses the problem of coordinating planning agents in naturally decentralized domains. Space missions will increasingly need technology for this as there is a trend toward larger multi-spacecraft missions and greater interaction of separate missions. We show how a Mars 2003 scenario involving only three spacecraft can benefit from decentralized coordination of planners. Our approach describes a simple, general, decentralized protocol based on communicating shared tasks. By specifying heterogeneous interpretations of shared tasks across agents and by limiting the set of tasks that each shares with others, the constraints associated with them, and the ability to modify certain kinds of shared tasks, a domain modeler can construct simple and complex coordination mechanisms seen in the coordination work of others. The protocol provides a layer of separation between the planning technology underneath and the coordination mechanisms built on top of it. This framework helps bridge the divide between decentralized coordination technology and planning systems and demonstrates its effectiveness in coordinating competitive space missions.

7. REFERENCES

[1] S. Chien, G. Rabideu, R. Knight, R. Sherwood, B. Engelhardt, D. Mutz, T. Estlin, B. Smith, F. Fisher, T. Barrett, G. Stebbins, and D. Tran. Automating space mission operations using automated planning and scheduling. In *Proc. SpaceOps*, 2000.

[2] B. Clement and E. Durfee. Performance of coordinating concurrent hierarchical planning agents using summary information. In *Proc. ATAL*, pages 202–216, 2000.

[3] D. Corkill. Heirarchical planning in a distributed environment. In *Proc. IJCAI*, pages 168–175, 1979.

[4] K. Decker. *Environment centered analysis and design of coordination mechanisms*. PhD thesis, University of Massachusetts, 1995.

[5] K. Decker and V. Lesser. Designing a family of coordination mechanisms. In *Proc. ICMAS*, pages 73–80, June 1995.

[6] M. des Jardins and M. Wolverton. Coordinating a distributed planning system. *AI Magazine*, 20(4):45–53, 1999.

[7] E. Ephrati and J. Rosenschein. Divide and conquer in multi-agent planning. In *Proc. AAAI*, pages 375–380, July 1994.

[8] M. P. Georgeff. Communication and interaction in multiagent planning. In *Proc. AAAI*, pages 125–129, 1983.

[9] B. J. Grosz and S. Kraus. Collaborative plans for complex group action. *Artificial Intelligence*, 86:269–358, 1996.

[10] H. Jung, M. Tambe, and S. Kulkami. Argumentation as distributed constraint satisfaction: Applications and results. In *Proc. Intl. Conf. Autonomous Agents*, 2001.

[11] R. Knight, G. Rabideau, and S. Chien. Extending the representational power of model-based systems using generalized timelines. ISAIRAS (abstract submitted), 2001.

[12] A. Lansky. Localized search for controlling automated reasoning. In *Proc. DARPA Workshop on Innov. Approaches to Planning, Scheduling and Control*, pages 115–125, November 1990.

[13] D. Pynadath, M. Tambe, N. Cauvat, and L. Cavedon. Toward team-oriented programming. In *Proc. ATAL*, 1999.

[14] A. S. Rao and M. P. Georgeff. BDI-agents: from theory to practice. In *Proceedings of the First Intl. Conference on Multiagent Systems*, San Francisco, 1995.

[15] M. Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7:83–124, 1997.

[16] M. Yokoo and K. Hirayama. The distributed constraint satisfaction problem: formalization and algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 10(5):673–685, 1998.